Sparse Algorithms are not Stable: A No-free-lunch Theorem

Huan Xu, Constantine Caramanis, *Member, IEEE* and Shie Mannor, *Senior Member, IEEE*

Abstract—We consider two desired properties of learning algorithms: *sparsity* and *algorithmic stability*. Both properties are believed to lead to good generalization ability. We show that these two properties contradict each other. That is, a sparse algorithm can not be stable and vice versa. Thus, one has to trade off sparsity and stability in designing a learning algorithm. In particular, our general result implies that ℓ_1-regularized regression (Lasso) cannot be stable, while ℓ_2-regularized regression is known to have strong stability properties and is therefore not sparse.

Index Terms—Stability, Sparsity, Lasso, Regularization

1 Introduction

Regression and classification are important problems in a broad range of applications. Given data points encoded by the rows of a matrix A, and observations or labels $\mathbf{b}$, the basic goal is to find a (linear) relationship between A and $\mathbf{b}$. Various objectives are possible, for example in regression, one may consider minimizing the least squared error, $||A\mathbf{w} - \mathbf{b}||_2$, or perhaps in case of a generative model assumption, minimizing the generalization error, i.e., the expected error of the regressor $\mathbf{w}$ on the next sample generated: $\mathbb{E}||\mathbf{a}^\top \mathbf{w} - b||$. In addition to such objectives, one may ask for solutions, $\mathbf{w}$, that have additional structural properties. In the machine learning literature, much work has focused on developing methodologies with special properties.

Two properties of particular interest are *sparsity* of the solution, and the *stability* of the algorithm. In a broad sense, stability means that an algorithm is well-posed, so that given two very similar data sets, an algorithm's output varies little. More specifically, an algorithm is stable if its output changes very little when given two data sets differing on only one sample (this is known as the leave-one-out error). Stability is by now a standard approach for establishing the generalization ability of learning algorithms following the landmark work of [1]. For example, in [2] the author uses stability properties of ℓ_2-regularized Support Vector Machine (SVM) to establish its consistency. Also see [3], [4], [5] and many others.

Similarly, numerous algorithms that encourage sparse solutions have been proposed in signal processing and virtually all fields in machine learning. A partial list includes: Lasso, 1-norm SVM, Deep Belief Network, Sparse PCA [6], [7], [8], [9], [10], [11] and many others. The popularity of algorithms that induce sparse solutions is due to the following reasons: (i) a sparse solution is less complicated and hence generalizes well [12]; (ii) a sparse solution has good interpretability [13], [14], [15], [16]; and (iii) sparse algorithms may be computationally much easier to implement, store, compress, etc.

In this paper, we investigate the mutual relationship of these two concepts. In particular, we show that sparse algorithms are not stable: if an algorithm "encourages sparsity" (in a sense defined precisely below) then its sensitivity to small perturbations of the input data remains bounded away from zero, i.e., it has no uniform stability properties. We define these notions formally in Section 2. We prove this "no-free-lunch" theorem by constructing an instance where the leave-one-out error of the algorithm is bounded away from zero by exploiting the property that a sparse algorithm can have non-unique optimal solutions, and is therefore *ill-posed*.

This paper is organized as follows. We start with the necessary definitions in Section 2 and provide the no-free-lunch theorem based on these definitions in Section 3. Sections 2 and 3 are devoted to regression algorithms; and in Section 4 we generalize the theorem to arbitrary loss functions. In Section 5 we discuss the justification of the particular notions of stability and sparsity considered in this paper. Brief concluding remarks are given in Section 6.

Notations: Capital letters (e.g., A) and boldface letters (e.g., $\mathbf{w}$) are used to denote matrices and column vectors, respectively. We use the transpose of a column vector to represent a row vector. Unless otherwise specified, the same letter is used to represent a part of an object. For example, the i^{th} column of a matrix A is denoted by $\mathbf{a}_i$. Similarly, the i^{th} element

- *H. Xu and C. Caramanis are with the Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX.*
  E-mail: huan.xu@mail.utexas.edu; caramanis@mail.utexas.edu.
- *S. Mannor is with the Department of Electrical Engineering, Technion, Haifa, ISRAEL.*
  E-mail: shie@ee.technion.ac.il

of a vector $\mathbf{d}$ is denoted by d_i.

2 Setup and Assumptions

We consider regression algorithms that find a weight vector, $\mathbf{w}^*$ in the input space. The goal of any algorithm we consider is to minimize the loss given a new observation $(\hat{b}, \hat{\mathbf{a}})$. Initially we consider the loss function $l(\mathbf{w}^*, (\hat{b}, \hat{\mathbf{a}})) = |\hat{b} - \hat{\mathbf{a}}^\top \mathbf{w}^*|$. Here, $\mathbf{a}$ is the vector of input values of the observation and $\hat{b}$ is the output . In the standard regression problem, the learning algorithm $\mathbb{L}$ obtains the candidate solution $\mathbf{w}^*$ by minimizing the empirical loss $||A\mathbf{w} - \mathbf{b}||_2$, or the regularized empirical loss. For a given objective function, we can compare two solutions $\mathbf{w}^1, \mathbf{w}^2$ by considering their empirical loss. We adopt a somewhat more general framework, considering only the partial ordering induced by any learning algorithm $\mathbb{L}$ and training set $(\mathbf{b}, A)$. That is, given two candidate solutions, $\mathbf{w}^1, \mathbf{w}^2$, we write

$$\mathbf{w}^1 \preceq_{(\mathbf{b},A)} \mathbf{w}^2,$$

if on input $(\mathbf{b}, A)$, the algorithm $\mathbb{L}$ would select $\mathbf{w}^2$ before $\mathbf{w}^1$. In short, given an algorithm $\mathbb{L}$, each sample set $(\mathbf{b}, A)$ defines an order relationship $\preceq_{(\mathbf{b},A)}$ among all candidate solutions $\mathbf{w}$. This order relationship defines a family of "best" solutions, and one of these, $\mathbf{w}^*$ is the output of the algorithm. We denote this by writing $\mathbf{w}^* \in \mathbb{L}_{(\mathbf{b},A)}$.

Thus, by defining a data-dependent partial order on the space of solutions, we can talk more generally about algorithms, their stability, and their sparsity. As we define below, an algorithm $\mathbb{L}$ is sparse if the set $\mathbb{L}_{(\mathbf{b},A)}$ of optimal solutions contains a sparse solution, and an algorithm is stable if the sets $\mathbb{L}_{(\mathbf{b},A)}$ and $\mathbb{L}_{(\hat{\mathbf{b}},\hat{A})}$ do not contain solutions that are very far apart, when $(\mathbf{b}, A)$ and $(\hat{\mathbf{b}}, \hat{A})$ differ on only one point.

We make a few assumptions on the preference order:

Assumption 1: (i) Given j, $\mathbf{b}$, A, $\mathbf{w}^1$ and $\mathbf{w}^2$, suppose that

$$\mathbf{w}^1 \preceq_{(\mathbf{b},A)} \mathbf{w}^2,$$

and

$$w_j^1 = w_j^2 = 0.$$

Then for any $\hat{\mathbf{a}}$,

$$\mathbf{w}^1 \preceq_{(\mathbf{b},\hat{A})} \mathbf{w}^2,$$

where

$$\hat{A} = (\mathbf{a}_1, \cdots, \mathbf{a}_{j-1}, \hat{\mathbf{a}}, \mathbf{a}_{j+1}, \cdots, \mathbf{a}_m).$$

(ii) Given $\mathbf{b}$, A, $\mathbf{w}^1$, $\mathbf{w}^2$, b' and $\mathbf{z}$, suppose that

$$\mathbf{w}^1 \preceq_{(\mathbf{b},A)} \mathbf{w}^2,$$

and

$$b' = \mathbf{z}^\top \mathbf{w}^2.$$

Then

$$\mathbf{w}^1 \preceq_{(\overline{\mathbf{b}},\overline{A})} \mathbf{w}^2,$$

where

$$\overline{\mathbf{b}} = \begin{pmatrix} \mathbf{b} \\ b' \end{pmatrix}; \quad \overline{A} = \begin{pmatrix} A \\ \mathbf{z}^\top \end{pmatrix}.$$

(iii) Given j, $\mathbf{b}$, A, $\mathbf{w}^1$ and $\mathbf{w}^2$, suppose that

$$\mathbf{w}^1 \preceq_{(\mathbf{b},A)} \mathbf{w}^2.$$

Then

$$\hat{\mathbf{w}}^1 \preceq_{(\mathbf{b},\tilde{A})} \hat{\mathbf{w}}^2,$$

where

$$\hat{\mathbf{w}}^i = \begin{pmatrix} \mathbf{w}^i \\ 0 \end{pmatrix}, \; i = 1, 2; \quad \tilde{A} = (A, \mathbf{0}).$$

(iv) Given $\mathbf{b}$, A, $\mathbf{w}^1$, $\mathbf{w}^2$ and $P \in \mathbb{R}^{m \times m}$ a permutation matrix, if

$$\mathbf{w}^1 \preceq_{(\mathbf{b},A)} \mathbf{w}^2,$$

then

$$P^\top \mathbf{w}^1 \preceq_{(\mathbf{b},AP)} P^\top \mathbf{w}^2.$$

Part (i) of the assumption says that the value of a column corresponding to a non-selected feature has no effect on the ordering. Part (ii) says that adding a sample that is perfectly predicted by a particular solution, cannot decrease its place in the partial ordering. Part (iii) says the order relationship is preserved when a trivial (all zeros) feature is added. Part (iv) says that the partial ordering and hence the algorithm, is feature-wise symmetric. These assumptions are intuitively appealing and satisfied by algorithms including, for instance, standard regression, and regularized regression. See Section 5 for additional examples that satisfy such assumptions.

In what follows, we will define precisely what we mean by stability and sparsity. We recall the definition of uniform (algorithmic) stability first, as given in [1]. We let $\mathcal{Z}$ denote the space of points and labels (typically this will either be $\mathbb{R}^{m+1}$ or a closed subset of it) so that $S \in \mathcal{Z}^n$ denotes a collection of n labelled training points. For regression problems, therefore, we have $S = (\mathbf{b}, A) \in \mathcal{Z}^n$. We let $\mathbb{L}$ denote a learning algorithm, and for $(\mathbf{b}, A) \in \mathcal{Z}^n$, we let $\mathbb{L}_{(\mathbf{b},A)}$ denote the output of the learning algorithm (i.e., the regression function it has learned from the training data). Then given a loss function l, and a labelled point $s = (b, \mathbf{z}) \in \mathcal{Z}$, $l(\mathbb{L}_{(\mathbf{b},A)}, s)$ denotes the loss of the algorithm that has been trained on the set $(\mathbf{b}, A)$, on the data point s. Thus in the regression setup, we would have $l(\mathbb{L}_{(\mathbf{b},A)}, s) = |\mathbb{L}_{(\mathbf{b},A)}(\mathbf{z}) - b|$.

Definition 1: [1] An algorithm $\mathbb{L}$ has uniform stability β_n with respect to the loss function l if the following holds:

$$\forall (\mathbf{b}, A) \in \mathcal{Z}^n, \forall i \in \{1, \cdots, n\}:$$
$$\max_{\mathbf{z}' \in \mathcal{Z}} |l(\mathbb{L}_{(\mathbf{b},A)}, \mathbf{z}') - l(\mathbb{L}_{(\mathbf{b},A)^{\setminus i}}, \mathbf{z}')| \leq \beta_n.$$

Here $\mathbb{L}_{(\mathbf{b},A)\setminus i}$ stands for the learned solution with the i^{th} sample removed from $(\mathbf{b}, A)$, i.e., with the i^{th} row of A and the i^{th} element of $\mathbf{b}$ removed.

At first glance, this definition may seem too stringent for any reasonable algorithm to exhibit good stability properties. However, as shown in [1], many algorithms have uniform stability with β_n going to zero. In particular, Tikhonov regularized regression (i.e., ℓ_2-regularized regression) has stability that goes to zero as $1/n$. Indeed, a recent work [17] shows that for $p > 1$, ℓ_p regularization has uniform stability with β_n going to zero as $1/n$. Stability can be used to establish strong PAC bounds. For example, [1] show that if we have n samples, β_n denotes the uniform stability, and M a bound on the loss, then with probability at least $1 - \delta$ the following hold,

$$R \le R_{\text{emp}} + 2\beta_n + (4n\beta_n + M)\sqrt{\frac{\ln 1/\delta}{2n}},$$

where R denotes the expected loss, and R_{emp} the empirical (i.e., training) loss.

Since Lasso is an example of an algorithm that yields sparse solutions, one implication of the results of this paper is that while ℓ_p-regularized ($p > 1$) regression yields stable solutions, ℓ_1-regularized regression does not. We show that the stability parameter of Lasso does not decrease in the number of samples (compared to the $O(1/n)$ decay for ℓ_p-regularized regression). In fact, we show that Lasso's stability is, in the following sense, the worst possible stability. To this end, we define the notion of the Pseudo Maximal Error (PME), which is the worst possible error a training algorithm can have for arbitrary training set and testing sample labelled by zero.

Definition 2: Given the sample space $\mathcal{Z} = \mathcal{Y} \times \mathcal{X}$ where $\mathcal{Y} \subseteq \mathbb{R}$, $\mathcal{X} \subseteq \mathbb{R}^m$, and $0 \in \mathcal{Y}$. The pseudo maximal error for a learning algorithm $\mathbb{L}$ w.r.t. $\mathcal{Z}$ is

$$\mathfrak{b}_n(\mathbb{L}, \mathcal{Z}) \triangleq \max_{(\mathbf{b},A)\in\mathcal{Z}^n, \mathbf{z}\in\mathcal{X}} l\big(\mathbb{L}_{(\mathbf{b},A)}, (0, \mathbf{z})\big).$$

As above, $l(\cdot, \cdot)$ is a given loss function.

As an example, if $\mathcal{X}$ is the unit ball, and $W(\mathbb{L})$ is the set of vectors $\mathbf{w}$ that are optimal with respect to at least one training set, then $\mathfrak{b}_n(\mathbb{L}, \mathcal{Z}) = \max_{\mathbf{w}\in W(\mathbb{L})} \|\mathbf{w}\|$. Thus, unless $\mathbb{L}$ is a trivial algorithm which always outputs $\mathbf{0}$, the PME is bounded away from zero.

Observe that $\mathfrak{b}_n(\mathbb{L}, \mathcal{Z}) \ge \mathfrak{b}_1(\mathbb{L}, \mathcal{Z})$, since by repeatedly choosing the worst sample (for $\mathfrak{b}_1$), the algorithm will yield the same solution. Hence the PME does not diminish as the number of samples, n, increases.

We next define the notion of sparsity of an algorithm which we use.

Definition 3: A weight vector $\mathbf{w}^*$ *Identifies Redundant Features of A* if

$$\forall i \neq j, \quad \mathbf{a}_i = \mathbf{a}_j \Rightarrow w_i^* w_j^* = 0.$$

An algorithm $\mathbb{L}$ is said to be *able to Identify Redundant Features* (IRF for short) if $\forall(\mathbf{b}, A)$ there exists $\mathbf{w}^* \in \mathbb{L}_{(\mathbf{b},A)}$ that identifies redundant features of A.

Being IRF means that at least one solution of the algorithm does not select both features if they are identical. We note that this is a quite weak notion of sparsity. An algorithm that achieves reasonable sparsity (such as Lasso) should be IRF. Notice that IRF is a property that is typically easy to check.

Before concluding this section, we comment on the two definitions that we considered, namely, the uniform stability and IRF.

The notion of *uniform stability* is arguably the most widely applied stability notion. More importantly, it does not involve the *unknown* generating distribution and is thus easy to evaluate, which makes it convenient to derive generalization bounds of learning algorithms. There are other notions of stability proposed in literature [3], [5]. Although these notions are less restrictive than the uniform stability, they often require knowledge of the distribution that generates samples. For example, [5] proposed a stability notion termed **all-i-LOO** stable, which requires that

$$\forall i \in \{1, \cdots, n\}: \quad \mathbb{E}_{S\sim\mu^n}|l(\mathbb{L}_S, s_i) - l(\mathbb{L}_{S^{\setminus i}}, s_i)| \le \beta_n,$$

where μ is the generating distribution. Because of the explicit dependence on μ, the all-i-LOO stability seems hard to evaluate.

The notion of IRF is proposed as an easily verifiable property that sparse algorithms should satisfy. While there are different notions of sparsity proposed in literature, the most widely applied notion of sparsity, recently popularized in the compressed sensing literature (and around in many, many other places) says that the sparsity of a vector is the number of nonzero elements, and an algorithm is sparse if it tends to find the most-sparse solution satisfying required performance (e.g., the regression error is sufficiently small). Under this definition, it is clear that IRF is a necessary property for an algorithm to be sparse.

3 The Main Theorem

The next theorem is the main contribution of this paper. It says that if an algorithm is sparse, in the sense that it identifies redundant features as in the definition above, then that algorithm *is not stable*. One notable example that satisfies this theorem is Lasso.

Theorem 1: Let $\mathcal{Z} = \mathcal{Y} \times \mathcal{X}$ be the sample space with m features, where $\mathcal{Y} \subseteq \mathbb{R}$, $\mathcal{X} \subseteq \mathbb{R}^m$, $0 \in \mathcal{Y}$ and $\mathbf{0} \in \mathcal{X}$. Let $\hat{\mathcal{Z}} = \mathcal{Y} \times \mathcal{X} \times \mathcal{X}$ be the sample space with $2m$ features. If a learning algorithm $\mathbb{L}$ (trained on points in $\hat{\mathcal{Z}}$) satisfies Assumption 1 and identifies redundant features, its uniform stability bound β is lower bounded by $\mathfrak{b}_n(\mathbb{L}, \mathcal{Z})$, and in particular does not go to zero with n.

Proof: Note that in light of the definition of uniform stability, it suffices to provide one example that algorithm $\mathbb{L}$ fails to achieve a small stability bound. We construct such an (somewhat extreme) example as follows.

Let $(\mathbf{b}, A)$ and $(0, \mathbf{z}^\top)$ be the sample set and the new observation such that they jointly achieve $\mathfrak{b}_n(\mathbb{L}, \mathcal{Z})$, i.e., for some $\mathbf{w}^* \in \mathbb{L}(\mathbf{b}, A)$, we have

$$\mathfrak{b}_n(\mathbb{L}, \mathcal{Z}) = l\big(\mathbf{w}^*, (0, \mathbf{z})\big). \tag{1}$$

Let $0^{n\times m}$ be the $n \times m$ 0-matrix, and $\mathbf{0}$ stand for the zero vector of length m. We denote

$$\hat{\mathbf{z}} \triangleq (\mathbf{0}^\top,\, \mathbf{z}^\top); \qquad \hat{A} \triangleq (A,\, A);$$
$$\tilde{\mathbf{b}} \triangleq \begin{pmatrix} \mathbf{b} \\ 0 \end{pmatrix}; \qquad \tilde{A} \triangleq \begin{pmatrix} A, & A \\ \mathbf{0}^\top, & \mathbf{z}^\top \end{pmatrix}.$$

Observe that $(\mathbf{b}, \hat{A}) \in \hat{\mathcal{Z}}^n$ and $(\tilde{\mathbf{b}}, \tilde{A}) \in \hat{\mathcal{Z}}^{n+1}$. We first show that

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{w}^* \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}; \quad \begin{pmatrix} \mathbf{w}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}. \tag{2}$$

Notice that $\mathbb{L}$ is feature-wise symmetric (Assumption 1(iv)) and I.R.F., hence there exists a $\mathbf{w}'$ such that

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{w}' \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}.$$

Since $\mathbf{w}^* \in \mathbb{L}_{(\mathbf{b},A)}$, we have

$$\begin{aligned}
& \mathbf{w}' \preceq_{(\mathbf{b},A)} \mathbf{w}^* \\
\Rightarrow\ & \begin{pmatrix} \mathbf{0} \\ \mathbf{w}' \end{pmatrix} \preceq_{(\mathbf{b},(0^{n\times m},A))} \begin{pmatrix} \mathbf{0} \\ \mathbf{w}^* \end{pmatrix} \\
\Rightarrow\ & \begin{pmatrix} \mathbf{0} \\ \mathbf{w}' \end{pmatrix} \preceq_{(\mathbf{b},\hat{A})} \begin{pmatrix} \mathbf{0} \\ \mathbf{w}^* \end{pmatrix} \\
\Rightarrow\ & \begin{pmatrix} \mathbf{0} \\ \mathbf{w}^* \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}.
\end{aligned}$$

The first implication follows from Assumption 1(iii), and the second from (i).

By Assumption 1(iv) (i.e., feature-wise symmetry), we have

$$\begin{pmatrix} \mathbf{w}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\mathbf{b},\hat{A})}.$$

Furthermore,

$$0 = (\mathbf{0}^\top, \mathbf{z}^\top) \begin{pmatrix} \mathbf{w}^* \\ \mathbf{0} \end{pmatrix},$$

and thus by Assumption 1(ii) we have

$$\begin{pmatrix} \mathbf{w}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}.$$

Hence (2) holds. This leads to (recall that $l(\mathbf{w}^*, (\hat{b}, \hat{\mathbf{a}})) = |\hat{b} - \hat{\mathbf{a}}^\top \mathbf{w}^*|$)

$$l\big(\mathbb{L}_{(\mathbf{b},\hat{A})}, (0, \hat{\mathbf{z}})\big) = l(\mathbf{w}^*, (0, \mathbf{z})); \quad l\big(\mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}, (0, \hat{\mathbf{z}})\big) = 0.$$

By definition of the uniform bound, we have

$$\beta \geq l\big(\mathbb{L}_{(\mathbf{b},\hat{A})}, (0, \hat{\mathbf{z}})\big) - l\big(\mathbb{L}_{(\tilde{\mathbf{b}},\tilde{A})}, (0, \hat{\mathbf{z}})\big).$$

Hence by (1) we have $\beta \geq \mathfrak{b}_n(\mathbb{L}, \mathcal{Z})$, which establishes the theorem. □

Theorem 1 not only means that a sparse algorithm is not stable, it also states that, if an algorithm is stable, there is no hope that it will be sparse, since it cannot even identify redundant features. For instance, ℓ_2 regularized regression is stable (see Example 3 with a linear kernel), and does not identify redundant features.

4 Generalization to Arbitrary Loss

So far our focus has been on the regression problem, i.e., the loss function is $l(\mathbf{w}^*, (\hat{b}, \hat{\mathbf{a}})) = |\hat{b} - \hat{\mathbf{a}}^\top \mathbf{w}^*|$. Of course, other loss functions may be of interest. For example, one may be interested in the ϵ-insensitive loss function $l(\mathbf{w}^*, (\hat{b}, \hat{\mathbf{a}})) = \max\big(|\hat{b} - \hat{\mathbf{a}}^\top \mathbf{w}^*| - \epsilon, 0\big)$ or the classification error $l(\mathbf{w}^*, (\hat{b}, \hat{\mathbf{a}})) = \mathbf{1}_{\hat{b}\neq \mathrm{sign}(\hat{\mathbf{a}}^\top \mathbf{w}^*)}$. Indeed, the results derived can easily be generalized to algorithms with arbitrary loss function having the form $l(\mathbf{w}^*, (\hat{b}, \hat{\mathbf{a}})) = f_m(\hat{b}, \hat{a}_1 w_i^*, \cdots, \hat{a}_m w_m^*)$ for some f_m (here, $\hat{a}_i$ and w_i^* denote the i^{th} component of $\hat{\mathbf{a}} \in \mathbb{R}^m$ and $\mathbf{w}^* \in \mathbb{R}^m$, respectively) that satisfies the following conditions:

$$\begin{aligned}
(a)\quad & f_m(b, v_1, \cdots, v_i, \cdots, v_j, \cdots v_m) \\
& = f_m(b, v_1, \cdots, v_j, \cdots, v_i, \cdots v_m);\ \forall b, \mathbf{v}, i, j. \\
(b)\quad & f_m(b, v_1, \cdots, v_m) = f_{m+1}(b, v_1, \cdots, v_m, 0);\ \forall b, \mathbf{v}.
\end{aligned} \tag{3}$$

In words, (a) means that the loss function is feature-wise symmetric, and (b) means that a dummy feature does not change the loss. Observe that both the $\epsilon-$insensitive loss and the classification error satisfy these conditions.

In contrast to the regression setup, under an arbitrary loss function, there may not exist a sample that can be perfectly predicted by the zero vector, which implies that Definition 2 can be overly stringent. We require following modification of Definition 2. The new definition thus also applies to the case where the sample domain does not contain examples with zero label.

Definition 4: Given $\mathcal{Z} = \mathcal{Y} \times \mathcal{X}$ where $\mathcal{Y} \subseteq \mathbb{R}$ and $\mathcal{X} \subseteq \mathbb{R}^m$, the pseudo maximal error for a learning algorithm $\mathbb{L}$ w.r.t. $\mathcal{Z}$

$$\hat{\mathfrak{b}}_n(\mathbb{L}, \mathcal{Z}) \triangleq \max_{(\mathbf{b},A)\in\mathcal{Z}^n, (b,\mathbf{z})\in\mathcal{Z}} \Big\{ l\big(\mathbb{L}_{(\mathbf{b},A)}, (b, \mathbf{z})\big) - l\big(\mathbf{0}, (b, \mathbf{z})\big) \Big\}.$$

The PME in the arbitrary loss case is thus defined as the largest (w.r.t. all possible testing samples) performance gap of outputs of a learning algorithm and the zero vector. Observe that Definition 4 is a relaxation of Definition 2 in the sense that if the loss function is indeed the regression error, then the PME defined by Definition 4 is larger than or equal to (i.e., *more unstable*) that of Definition 2.

To account for the modification of Definition 2, we need to make Assumption 1 slightly stronger: we replace Assumption 1(ii) with the following one.

Assumption 2: (ii) Given $\mathbf{b}$, A, $\mathbf{w}^1$, $\mathbf{w}^2$, b' and $\mathbf{z}$ if

$$\mathbf{w}^1 \preceq_{(\mathbf{b},A)} \mathbf{w}^2, \quad l(\mathbf{w}^2, (b', \mathbf{z})) \le l(\mathbf{w}^1, (b', \mathbf{z}))$$

then

$$\mathbf{w}^1 \preceq_{(\overline{\mathbf{b}},\overline{A})} \mathbf{w}^2, \quad \text{where } \overline{\mathbf{b}} = \begin{pmatrix} \mathbf{b} \\ b' \end{pmatrix}; \quad \overline{A} = \begin{pmatrix} A \\ \mathbf{z}^\top \end{pmatrix}.$$

Assumption 2(ii) means that adding a sample that is better predicted (i.e., smaller loss) can not make a candidate solution less preferred.

With these modifications, we have a generalization of Theorem 1. The proof is similar to that of Theorem 1 and hence deferred to Appendix A.

Theorem 2: Let $\mathcal{Z} = \mathcal{Y} \times \mathcal{X}$ be the sample space with m features, where $\mathcal{Y} \subseteq \mathbb{R}$, $\mathcal{X} \subseteq \mathbb{R}^m$, and $\mathbf{0} \in \mathcal{X}$. Let $\hat{\mathcal{Z}} = \mathcal{Y} \times \mathcal{X} \times \mathcal{X}$ be the sample space with $2m$ features. If a learning algorithm $\mathbb{L}$ (trained on points in $\hat{\mathcal{Z}}$) satisfies Assumption 2 and identifies redundant features, its uniform stability bound β is lower bounded by $\hat{\mathfrak{b}}_n(\mathbb{L}, \mathcal{Z})$, and in particular does not go to zero with n.

While this paper focuses on the case where a learned solution takes a vector form, it is straightforward to generalize the setup to the matrix case and show that a similar no-free-lunch theorem between stability and group sparsity holds. As an example, consider the following group-sparse algorithm: $\text{Minimize:}_W \|B - AW\|_F + \|W\|_{1,2}$; where $\|W\|_{1,2}$ is the summation of the ℓ_2 norm of each row of W. Then, treating *each row* of W as the value of a feature of the solution and following a similar argument as the proof of Theorem 1, one can show that such a group sparse algorithm is not stable. Due to space constraint, we do not elaborate.

5 Discussion

To see that the two notions of stability and sparsity that we consider are not too restrictive, we list in this section some algorithms that either admit a diminishing uniform stability bound or identify redundant features. Thus, by applying Theorem 2 we conclude that they are either non-sparse or non-stable.

5.1 Stable algorithms

All algorithms listed in this section have a uniform stability bound that decreases as $O(\frac{1}{n})$, and are hence stable. Examples 1 to 5 and adapted from [1].

Example 1 (Bounded SVM regression): Assume k is a bounded kernel, that is $k(\mathbf{x}, \mathbf{x}) \le \kappa^2$. Let $\mathcal{F}$ denote the RKHS space of k. Consider $\mathcal{Y} = [0,\, B]$ and the loss function

$$\begin{aligned} l(f, (y, \mathbf{x})) &= |f(\mathbf{x}) - y|_\epsilon \\ &= \begin{cases} 0 & \text{if } |f(\mathbf{x}) - y| \le \epsilon; \\ |f(\mathbf{x}) - y| - \epsilon & \text{otherwise.} \end{cases} \end{aligned}$$

The SVM regression algorithm with kernel k is defined as

$$\mathbb{L}_S = \arg\min_{g \in \mathcal{F}} \Big\{ \sum_{i=1}^n l(g, (y_i, \mathbf{x}_i)) + \lambda n \|g\|_\kappa^2 \Big\};$$

where, $S = ((y_1, \mathbf{x}_1), \cdots, (y_n, \mathbf{x}_n))$. Then, its uniform stability satisfies

$$\beta_n \le \frac{\kappa^2}{2\lambda n}.$$

Example 2 (Soft-margin SVM classification): Assume k is a bounded kernel, that is $k(\mathbf{x}, \mathbf{x}) \le \kappa^2$. Let $\mathcal{F}$ denote the RKHS space of k. Consider $\mathcal{Y} = \{0,\, 1\}$[1] and the loss function

$$\begin{aligned} &l(f, (y, \mathbf{x})) = (1 - (2y-1)f(\mathbf{x}))^+ \\ &= \begin{cases} 1 - (2y-1)f(\mathbf{x}) & \text{if } 1 - (2y-1)f(\mathbf{x}) > 0; \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

The soft-margin SVM (without bias) algorithm with kernel k is defined as

$$\mathbb{L}_S = \arg\min_{g \in \mathcal{F}} \Big\{ \sum_{i=1}^n l(g, (y_i, \mathbf{x}_i)) + \lambda n \|g\|_\kappa^2 \Big\};$$

where $S = ((y_1, \mathbf{x}_1), \cdots, (y_n, \mathbf{x}_n))$. Then, its uniform stability satisfies

$$\beta_n \le \frac{\kappa^2}{2\lambda n}.$$

Example 3 (RKHS regularized least square regression): Assume k is a bounded kernel, that is $k(\mathbf{x}, \mathbf{x}) \le \kappa^2$. Let $\mathcal{F}$ denote the RKHS space of k. Consider $\mathcal{Y} = [0,\, B]$ and the loss function

$$l(f, (y, \mathbf{x})) = (f(\mathbf{x}) - y)^2.$$

The regularized least square regression algorithm with kernel k is defined as

$$\mathbb{L}_S = \arg\min_{g \in \mathcal{F}} \Big\{ \sum_{i=1}^n l(g, (y_i, \mathbf{x}_i)) + \lambda n \|g\|_\kappa^2 \Big\};$$

where: $S = ((y_1, \mathbf{x}_1), \cdots, (y_n, \mathbf{x}_n))$. Then, its uniform stability satisfies

$$\beta_n \le \frac{2\kappa^2 B^2}{\lambda n}.$$

The next example is relative entropy regularization. In this case, we are given a class of base hypotheses, and the output of the algorithm is a mixture of them, or more precisely a probability distribution over the class of base hypotheses.

Example 4 (Relative Entropy Regularization): Let $\mathcal{H} = \{h_\theta : \theta \in \Theta\}$ be the class of base hypotheses, where Θ is a measurable space with a reference measure. Let $\mathcal{F}$ denote the set of probability distributions over Θ dominated by the reference measure. Consider the loss function for $f \in \mathcal{F}$

$$l(f, \mathbf{z}) = \int_\Theta r(h_\theta, \mathbf{z}) f(\theta) d\theta;$$

[1] This is slightly different from but equivalent to the standard setup where $\mathcal{Y} = \{-1,\, 1\}$.

where $r(\cdot,\cdot)$ is a loss function bounded by M. Further, let f_0 be a fixed element of $\mathcal{F}$ and $K(\cdot,\cdot)$ denote the Kullback-Leibler divergence. The relative entropy regularized algorithm is defined as

$$\mathbb{L}_S = \arg\min_{g\in\mathcal{F}} \Big\{ \sum_{i=1}^{n} l(g,\mathbf{z}_i) + \lambda n K(g, f_0)\Big\};$$

where $S = (\mathbf{z}_1, \cdots, \mathbf{z}_n)$. Then, its uniform stability satisfies

$$\beta_n \leq \frac{M^2}{\lambda n}.$$

A special case of relative entropy regularization is the following *maximum entropy discrimination* proposed in [18].

Example 5 (Maximum entropy discrimination): Let $\mathcal{H} = \{h_{\theta,\gamma} : \theta \in \Theta, \gamma \in \mathbb{R}\}$. Let $\mathcal{F}$ denote the set of probability distributions over $\Theta \times \mathbb{R}$ dominated by the reference measure. Consider $\mathcal{Y} = \{0, 1\}$ and the loss function

$$l(f,\mathbf{z}) = \left(\int_{\Theta,\mathbb{R}} [\gamma - (2y-1)h_{\theta,\gamma}(\mathbf{x})] f(\theta,\gamma) d\theta d\gamma\right)_+;$$

where $[\gamma - (2y-1)h_{\theta,\gamma}(\mathbf{x})]$ is bounded by M. The maximum entropy discrimination is a real-valued classifier defined as

$$\mathbb{L}_S = \arg\min_{g\in\mathcal{F}} \Big\{ \sum_{i=1}^{n} l(g,\mathbf{z}_i) + \lambda n K(g, f_0)\Big\};$$

where $S = (\mathbf{z}_1, \cdots, \mathbf{z}_n)$. Then, its uniform stability satisfies

$$\beta_n \leq \frac{M}{\lambda n}.$$

If an algorithm is not stable, one way to stabilize it is to average its solutions trained on small bootstrap subsets of the training set, a process called subbagging [19], which we recall in the following example.

Example 6 (Subbagging, see Theorem 5.2 of [19].): Let $\mathbb{L}$ be a learning algorithm with a stability β_n, and consider the following algorithm

$$\hat{\mathbb{L}}^k_{\mathcal{D}}(\mathbf{x}) \triangleq \mathbb{E}_{\mathcal{S}}\left(\mathbb{L}_{\mathcal{S}}(\mathbf{x})\right).$$

where $\mathbb{E}_{\mathcal{S}}$ is the expectation with respect to k points sampled in $\mathcal{D}$ uniformly *without replacement*. Then $\hat{\mathbb{L}}^k$ has a stability $\hat{\beta}_n$ satisfying

$$\hat{\beta}_n \leq \frac{k}{n}\beta_k.$$

In a recent work, [17] consider the uniform stability of ℓ_p regularization for $1 < p \leq 2$ and elastic net proposed in [20]. Their results imply the following examples.

Example 7 (ℓ_p regularization): Consider a collection of feature functions $(\varphi_\gamma(\cdot))_{\gamma\in\Gamma}$, where Γ is a countable set, such that for every $\mathbf{x} \in \mathcal{X}$,

$$\sum_{\gamma\in\Gamma} |\varphi_\gamma(\mathbf{x})|^2 \leq \kappa.$$

Let $\mathcal{F}$ denote the linear span of the feature functions, i.e.,

$$\mathcal{F} = \{\sum_{\gamma\in\Gamma} \alpha_\gamma \varphi_\gamma(\cdot) : \boldsymbol{\alpha} \in \ell_2(\Gamma)\}.$$

Further assume that the loss function is such that $l(f, (y, \mathbf{x})) = V(f(\mathbf{x}), y)$, for some $V(\cdot,\cdot)$ that is convex, bounded, and Lipschitz continuous. That is, $V(\cdot,\cdot)$ satisfies

1) V is convex.
2) For all y, y' we have $0 \leq V(y', y) \leq B$.
3) For all y_1, y_2, y, we have $|V(y_1, y) - V(y_2, y)| \leq L|y_1 - y_2|$.

Then, the ℓ_p regularization algorithm, defined as

$$\begin{aligned}\mathbb{L}_S \;=\; \arg\min_{\boldsymbol{\alpha}\in\ell_2(\Gamma)} \Big\{ &\sum_{i=1}^{n} l(\sum_{\gamma\in\Gamma} \alpha_\gamma\varphi_\gamma(\cdot), (y_i, \mathbf{x}_i)) \\ &+\lambda n \sum_{\gamma\in\Gamma} |\alpha_\gamma|^p \Big\};\end{aligned}$$

where $S = ((y_1, \mathbf{x}_1), \cdots, (y_n, \mathbf{x}_n))$, is uniformly stable with

$$\beta_n = \frac{1}{p(p-1)} \left(\frac{B}{\lambda}\right)^{(2-p)/p} \frac{4L^2\kappa}{n\lambda}.$$

Observe that up to a constant, Example 1 to 3 are special cases of Example 7 with $p = 2$. One interesting observation is that when $p = 1$ the stability bound breaks. As we know from previous sections, this is due to the sparsity of ℓ_1 regularization.

Example 8 (Elastic Net): Under the same assumptions as Example 7, the elastic-net regularization algorithm, defined as

$$\begin{aligned}\mathbb{L}_S \;=\; \arg\min_{\boldsymbol{\alpha}\in\ell_2(\Gamma)} \Big\{ &\sum_{i=1}^{n} l(\sum_{\gamma\in\Gamma} \alpha_\gamma\varphi_\gamma(\cdot), (y_i, \mathbf{x}_i)) \\ &+\lambda n \sum_{\gamma\in\Gamma} (w_\gamma|\alpha_\gamma| + \epsilon\alpha_\gamma^2) \Big\};\end{aligned}$$

where $S = ((y_1, \mathbf{x}_1), \cdots, (y_n, \mathbf{x}_n))$, for some $w_\gamma \geq 0$, is uniformly stable with

$$\beta_n = \frac{2L^2\kappa}{\epsilon n\lambda}.$$

Note that the weights $(w_\gamma)_{\gamma\in\Gamma}$ have no effect in the stability bound. This is easily expected as ℓ_1 regularization itself is not stable. Indeed, the stability bound of the elastic net coincides with that of a ℓ_2 regularization algorithm. One may easily check that because of the extra ℓ_2 norm, elastic nets do not enjoy the property of IRF.

We briefly comment on the last example, the *elastic net*. In [20] the authors proposed elastic net and used the terminology "sparsity," but the meaning seems to be quite different than ours. Motivated by biomedical applications, the authors of [20] are not interested in not spreading weight to multiple features if those features are similar or identical, indeed, they are aiming at the exact opposite: to spread out weight to multiple similar features. Clearly this is not the notion of "sparsity" we have (and many other papers are interested). The notion of sparsity we consider means ability to find the solution with fewest non-zero coefficients. [2] Therefore, this example does not contradict to our claim that sparse algorithms are not stable.

5.2 Sparse Algorithms

Next we list some algorithms that identify redundant features.

Example 9 (ℓ_0 Minimization): Subset selection algorithms based on minimizing ℓ_0 norm identify redundant features. One example of such an algorithm is the *canonical selection procedure* [21], which is defined as

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathbb{R}^m} \{\|A\mathbf{w}-\mathbf{b}\|_2 + \lambda\|\mathbf{w}\|_0\}. \qquad (4)$$

Proof: Note that if a solution $\mathbf{w}^*$ achieves the minimum of (4) and has non-zero weights on two redundant features i and i', then by constructing a $\hat{\mathbf{w}}$ such that $\hat{w}_i = w_i^* + w_{i'}^*$ and $\hat{w}_{i'} = 0$ we get a strictly better solution, which is a contradiction. Hence ℓ_0 minimizing algorithms is IRF. □

It is known that in general finding the minimum of (4) is NP-hard [22]. Therefore, a convex relaxation, the ℓ_1 norm, is used instead to find a sparse solution. These algorithms either minimize the ℓ_1 norm of the solution under the constraint of a regression error, or minimize the convex combination of some regression error and the ℓ_1 norm of the solution.

Example 10 (ℓ_1 Minimization): The following subset selection algorithms based on minimizing the ℓ_1 norm to identify redundant features. These algorithms include:

1) *Lasso* [6] defined as

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathbb{R}^m} \left\{\|A\mathbf{w}-\mathbf{b}\|_2^2 + \lambda\|\mathbf{w}\|_1\right\}.$$

And equivalently, the LARS algorithm [23] that solves Lasso.

2) *Basis Pursuit* [24] defined as the solution of the following optimization problem on $\mathbf{w}\in\mathbb{R}^m$:

$$\begin{aligned}\min &: \|\mathbf{w}\|_1\\ \text{s.t.} &: A\mathbf{w} = \mathbf{b}.\end{aligned}$$

3) *Dantzig Selector* [25] defined as

$$\begin{aligned}\text{Minimize:}\quad & \|\mathbf{w}\|_1\\ \text{Subject to:}\quad & \|A^*(A\mathbf{w}-\mathbf{b})\|_\infty \le c.\end{aligned}$$

Here, A^* is the complex conjugate of A, and c is some positive constant.

4) 1-norm SVM [7], [8] defined as the solution of the following optimization problem on $\boldsymbol{\alpha}$, $\boldsymbol{\xi}$, γ.

$$\begin{aligned}\min &: \|\boldsymbol{\alpha}\|_1 + C\sum_{i=1}^n \xi_i\\ \text{s.t.} &: y_i\Big\{\sum_{j=1}^n \alpha_i k(\mathbf{x}_i,\mathbf{x}_j)+\gamma\Big\} \ge 1-\xi_i;\ \forall i;\\ &\xi_i \ge 0;\quad \forall i.\end{aligned}$$

5) *ℓ_1 norm SVM regression* [26] defined as the solution of the following optimization problem on $\boldsymbol{\alpha}$, $\boldsymbol{\xi}$ and γ:

$$\begin{aligned}\min &: \|\boldsymbol{\alpha}\|_1 + C\sum_{i=1}^n \xi_i\\ \text{s.t.} &: \Big\{\sum_{j=1}^n \alpha_i k(\mathbf{x}_i,\mathbf{x}_j)+\gamma\Big\} - y_i \le \varepsilon + \xi_i;\ \forall i;\\ & y_i - \Big\{\sum_{j=1}^n \alpha_i k(\mathbf{x}_i,\mathbf{x}_j)+\gamma\Big\} \le \varepsilon + \xi_i;\ \forall i;\\ &\xi_i \ge 0;\quad \forall i,\end{aligned}$$

where $\varepsilon > 0$ is a fixed constant.

Proof: Given an optimal $\mathbf{w}^*$ we construct a new solution $\hat{\mathbf{w}}$ such that for any subset of redundant features I, $\sum_{i\in I}\mathbf{1}(\hat{w}_i \neq 0) \le 1$ and $\sum_{i\in I}\hat{w}_i = \sum_{i\in I} w_i^*$. Thus, $\hat{\mathbf{w}}$ and $\mathbf{w}^*$ are equally good, which implies that any ℓ_1 minimizing algorithm has at least one optimal solution that is IRF. Hence such algorithm is IRF by definition. □

6 Conclusion

In this paper, we prove that sparsity and stability are at odds with each other. We show that if an algorithm is sparse, then its uniform stability is lower bounded by a nonzero constant. This also shows that any algorithmically stable algorithm cannot be sparse. Thus, we show that these two widely used concepts, namely *sparsity* and *algorithmic stability* contradict each other. At a high level, this theorem provides us with additional insight into these concepts and their interrelation, and it furthermore implies that a tradeoff between these two concepts is unavoidable in designing learning algorithms. Given that both sparsity and stability are desirable properties, one interesting direction is to understand the full implications of having one of them. That is, what other properties must a sparse solution have? Given that sparse algorithms often perform well, one may further ask for meaningful and computable notions of stability that are not in conflict with sparsity.

Appendix A
Proof of Theorem 2:

Proof: This proof follows a similar line of reasoning as the proof of Theorem 1. Let $(\mathbf{b}, A)$ and

[2] Indeed, because of the extra ℓ_2 term, in almost all instances, the elastic net would output a solution with at least the same number of non-zero coefficients as the ℓ_1 regularization, and sometimes output a much denser solution.

$(b', \mathbf{z}^\top)$ be the sample set and the new observation such that they jointly achieve $\hat{\mathfrak{b}}_n(\mathbb{L}, \mathcal{Z})$, i.e., there exists $\mathbf{w}^* \in \mathbb{L}(\mathbf{b}, A)$ such that:

$$\begin{aligned}\hat{\mathfrak{b}}_n(\mathbb{L}, \mathcal{Z}) &= l\big(\mathbf{w}^*, (b', \mathbf{z})\big) - l\big(\mathbf{0}, (b',\, \mathbf{z})\big)\\ &= f_m(b', w_1^* z_1, \cdots, w_m^* z_m) - f(b', 0, \cdots, 0).\end{aligned}$$

Let $0^{n\times m}$ be the $n \times m$ 0-matrix, and $\mathbf{0}$ stand for the zero vector of length m. We denote

$$\hat{\mathbf{z}} \triangleq (\mathbf{0}^\top,\, \mathbf{z}^\top); \qquad \hat{A} \triangleq (A,\, A);$$
$$\tilde{\mathbf{b}} \triangleq \begin{pmatrix} \mathbf{b} \\ b' \end{pmatrix}; \qquad \tilde{A} \triangleq \begin{pmatrix} A, & A \\ \mathbf{0}^\top, & \mathbf{z}^\top \end{pmatrix}.$$

Observe that $(\mathbf{b}, \hat{A}) \in \hat{\mathcal{Z}}^n$ and $(\tilde{\mathbf{b}}, \tilde{A}) \in \hat{\mathcal{Z}}^{n+1}$. To prove the theorem, it suffices to show that there exist $\mathbf{w}^1$, $\mathbf{w}^2$ such that

$$\mathbf{w}^1 \in \mathbb{L}_{(\mathbf{b}, \hat{A})}, \quad \mathbf{w}^2 \in \mathbb{L}_{(\tilde{\mathbf{b}}, \tilde{A})},$$

and

$$l\big(\mathbf{w}^1, (b', \hat{\mathbf{z}})\big) - l\big(\mathbf{w}^2, (b', \hat{\mathbf{z}})\big) \geq \hat{\mathfrak{b}}_n(\mathbb{L}, \mathcal{Z})$$

where again,

$$\hat{\mathfrak{b}}_n(\mathbb{L}, \mathcal{Z}) = f_m(b', w_1^* z_1, \cdots, w_m^* z_m) - f_m(b', 0, \cdots, 0).$$

By an identical argument to the proof of Theorem 1, Assumption 1(i), (iii) and (iv) imply that:

$$\begin{pmatrix} \mathbf{0} \\ \mathbf{w}^* \end{pmatrix} \in \mathbb{L}_{(\mathbf{b}, \hat{A})}.$$

Hence there exists $\mathbf{w}^1 \in \mathbb{L}_{(\mathbf{b}, \hat{A})}$ such that

$$\begin{aligned} l\left(\mathbf{w}^1, (b', \hat{\mathbf{z}})\right) &= l\left(\begin{pmatrix} \mathbf{0} \\ \mathbf{w}^* \end{pmatrix}, (b', \hat{\mathbf{z}})\right) \\ &= f_m(b', w_1^* z_1, \cdots, w_m^* z_m). \end{aligned} \tag{5}$$

The last equality follows from Equation (3) easily. By feature-wise symmetry (Assumption 1(iv)), we have

$$\begin{pmatrix} \mathbf{w}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\mathbf{b}, \hat{A})}. \tag{6}$$

Hence there exists $\mathbf{w}^2 \in \mathbb{L}_{(\tilde{\mathbf{b}}, \tilde{A})}$ such that

$$\begin{aligned} l\left(\mathbf{w}^2, (b', \hat{\mathbf{z}})\right) &\leq l\left(\begin{pmatrix} \mathbf{w}^* \\ \mathbf{0} \end{pmatrix}, (b', \hat{\mathbf{z}})\right) \\ &= f_m(b', 0, \cdots, 0). \end{aligned} \tag{7}$$

The last equality follows from Equation (3). The inequality here holds because by Assumption 2(ii), if there is no $\mathbf{w}^2 \in \mathbb{L}_{(\tilde{\mathbf{b}}, \tilde{A})}$ that satisfies the inequality, then by (6) and definition of $\tilde{\mathbf{b}}$ and $\tilde{A}$ we have

$$\mathbf{w}^2 \preceq_{(\tilde{\mathbf{b}}, \tilde{A})} \begin{pmatrix} \mathbf{w}^* \\ \mathbf{0} \end{pmatrix}$$

which implies that

$$\begin{pmatrix} \mathbf{w}^* \\ \mathbf{0} \end{pmatrix} \in \mathbb{L}_{(\tilde{\mathbf{b}}, \tilde{A})},$$

from the optimality of $\mathbf{w}^2$. However, this is a contradiction of the assumption that there is no $\mathbf{w}^2 \in \mathbb{L}_{(\tilde{\mathbf{b}}, \tilde{A})}$ that satisfies the inequality of (7).

Combining (5) and (7) proves the theorem. □

REFERENCES

[1] O. Bousquet and A. Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2:499–526, 2002.

[2] I. Steinwart. Consistency of support vector machines and other regularized kernel classifiers. *IEEE Transactions on Information Theory*, 51(1):128–142, 2005.

[3] T. Poggio, R. Rifkin, S. Mukherjee, and P. Niyogi. General conditions for predictivity in learning theory. *Nature*, 428(6981):419–422, 2004.

[4] S. Mukherjee, P. Niyogi, T. Poggio, and R. Rifkin. Learning theory: Stability is sufficient for generalization and necessary and sufficient for consistency of empirical risk minimization. *Advances in Computational Mathematics*, 25(1-3):161–193, 2006.

[5] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. Learnability and stability in the general learning setting. In *Proceedings of 22nd Annual Conference of Learning Theory*, 2009.

[6] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society, Series B*, 58(1):267–288, 1996.

[7] O. L. Mangasarian. Generalized support vector machines. In A. J. Smola, P. L. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 135–146. MIT Press, 2000.

[8] J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. In *Advances in Neural Information Processing Systems 16*, 2003.

[9] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with nerual networks. *Science*, 313, 2006.

[10] A. d'Aspremont, L El Ghaoui, M. I. Jordan, and G. R. Lanckriet. A direct formulation for sparse PCA using semidefinite programming. *SIAM Review*, 49(3):434–448, 2007.

[11] A. d'Aspremont, F. Bach, and L. El Ghaoui. Full regularization path for sparse principal component analysis. In *Proceedings of International Conference on Machine Learning*, 2007.

[12] F. Girosi. An equivalence between sparse approximation and support vector machines. *Neural Computation*, 10(6):1445–1480, 1998.

[13] R. R. Coifman and M. V. Wickerhauser. Entropy-based algorithms for best basis selection. *IEEE Transactions on Information Theory*, 38(2):713–718, 1992.

[14] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

[15] E. J. Candès, J. Romberg, and T. Tao. Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *IEEE Transactions on Information Theory*, 52(2):489–509, 2006.

[16] D. L. Donoho. Compressed sensing. *IEEE Transactions on Information Theory*, 52(4):1289–1306, 2006.

[17] A. Wibisono, L. Rosasco, and T. Poggio. Sufficient conditions for uniform stability of regularization algorithms. Technical Report MIT-CSAIL-TR-2009-060, Massachusetts Institute of Technology, 2009.

[18] T. Jaakkola, M. Meila, and T. Jebara. Maximum entropy discrimination. In *Advances in Neural Information Processing Systems 12*, pages 470–476. MIT Press, 1999.

[19] T. Evgeniou, M. Pontil, and A. Elisseeff. Leave one out error, stability, and generalization of voting combinations of classifiers. *Machine Learning*, 55(1):71–97, 2004.

[20] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society Series B*, 67(2):301–320, 2005.

[21] D. P. Foster and E. I. George. The risk inflation criterion for multiple regression. *The Annals of Statistics*, 22:1947–1975, 1994.

[22] B. K. Natarajan. Sparse approximate solutions to linear systems. *SIAM Journal of Computation*, 24:227–234, 1995.

[23] B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *The Annals of Statistics*, 32(2):407–499, 2004.

[24] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Journal on Scientific Computing*, 20(1):33–61, 1999.

[25] E. J. Candès and T. Tao. The Dantzig selector: Statistical estimation when p is much larger than n. *The Annals of Statistics*, 35(6):2313–2351, 2007.

[26] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.